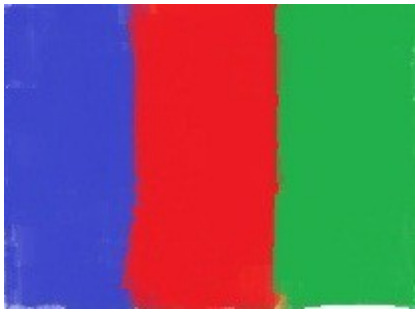


À Rabat le 02 Septembre 2026

PARAMÉTRISATION DE L'ÉNERGIE SOMBRE DANS L'ESPACE RACHID ELAÏDI (ERE) : UNE GÉNÉRALISATION NON-PERTURBATIVE AU-DELÀ DE LA PARAMÉTRISATION CPL

Extension du formalisme ERE pour une équation d'état de l'énergie sombre incluant des termes d'ordre supérieur, des énergies sporadiques et une séparation catégorique via l'algèbre de Grothendieck

Rachid ELAÏDI
Chercheur indépendant – Directeur de recherche
Physique fondamentale – Spécialité : monde visible et ERE (Espace Rachid ELAÏDI / monde invisible)
Plus de 150 publications – Dépôt sur ResearchGate
ORCID : 0000-0002-9599-4735

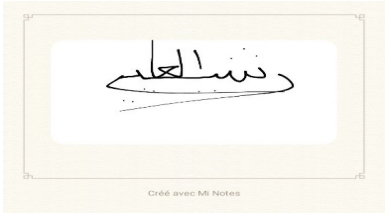
RÉSUMÉ STRUCTURÉ

Contexte. L'énergie sombre demeure l'une des plus grandes énigmes de la cosmologie moderne. La paramétrisation CPL (Chevallier-Polarski-Linder), largement utilisée pour modéliser son équation d'état $w(z) = w_0 + w_a z/(1+z)$, est une expansion de Taylor tronquée dont les termes d'ordre supérieur sont arbitrairement fixés à zéro. Une étude récente de Nesseris, Akrami et Starkman (Phys. Rev. D 114, L041305, 2026) démontre que cette troncature introduit un biais significatif, conduisant à des conclusions potentiellement erronées sur la nature dynamique ou non de l'énergie sombre.

Objectifs. Cet article propose une généralisation non-perturbative de la paramétrisation CPL dans le cadre du formalisme de l'Espace Rachid ELAÏDI (ERE). L'objectif est triple : (i) inclure naturellement les termes d'ordre supérieur via la série Rachid ELAÏDI (Série_RE), (ii) intégrer une double composante énergétique — énergie sombre programmée et énergies sporadiques —, et (iii) formaliser la séparation de ces composantes par l'algèbre de Grothendieck.

Méthodes. Nous définissons une **paramétrisation ERE** de l'équation d'état :

$w_ERE(z, \varphi, x) = w_0 + w_a \times [z/(1+z)] + Série_RE(\varphi, x) \times \Sigma_{r=2}^{\infty} w_r \times [z/(1+z)]^r$



où Série_RE est la série Rachid ELAÏDI, φ la phase modulo nπ, et w_r des coefficients programmés par la source P_source dans E_AvAvBB. L'énergie sombre programmée est définie par $E_dark^{(m)}(X) = E_0 \times Série_RE^{(m)} \times \exp(-\Theta_dark(t) \times |k|) \times (1 - T_m(X))$. Les énergies sporadiques sont un ensemble plus vaste incluant les énergies sombres (avec variations spécifiques) et des énergies encore inconnues. La séparation est formalisée par un topos de Grothendieck Sh(C, J) où C est l'arbre Rachid ELAÏDI, J la topologie de couverture, et F_dark, F_sporad des faisceaux séparés.

Résultats. La paramétrisation ERE est une généralisation directe de la paramétrisation CPL : CPL est retrouvée lorsque Série_RE → 1 et w_r = 0 pour r ≥ 2. Le formalisme résout les trois limites identifiées par Nesseris et al. : (i) élimination du biais par inclusion naturelle des termes d'ordre supérieur, (ii) correction du comportement fantôme artificiel, (iii) pas de conclusion erronée sur la constante cosmologique. La double composante énergétique permet de concilier une structure déterministe (programmation) avec des fluctuations stochastiques (énergies sporadiques). L'algèbre de Grothendieck garantit la séparation des faisceaux F_dark et F_sporad par une condition de recollement unique.

Conclusion. Le formalisme ERE offre une généralisation non-perturbative et programmable de la paramétrisation CPL, résolvant les limites identifiées par la communauté scientifique. Il intègre une énergie sombre programmée, des énergies sporadiques (incluant des

énergies inconnues), et une séparation catégorique rigoureuse. Des prédictions testables sont énoncées : déviations par rapport à CPL, corrélations avec les anomalies isotopiques, signatures dans le fond diffus cosmologique.

Mots-clés : Énergie sombre, paramétrisation CPL, équation d'état, Espace Rachid ELAÏDI (ERE), série Rachid ELAÏDI, programmation source, mondes jumeaux, énergies sporadiques, algèbre de Grothendieck, faisceaux, topos, cosmologie non-perturbative.

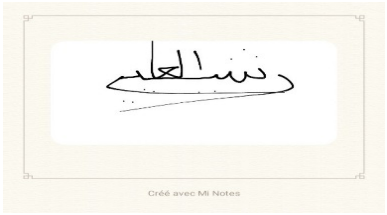
---

1. INTRODUCTION

1.1 La paramétrisation CPL et ses limites

L'énergie sombre est l'une des plus grandes énigmes de la cosmologie moderne. Depuis sa découverte en 1998 , son équation d'état, définie par w = p/ρ, reste mal connue. La paramétrisation la plus utilisée pour modéliser son évolution est la paramétrisation CPL (Chevallier-Polarski-Linder) :

$$w_{CPL}(z) = w_0 + w_a \times z/(1+z)$$



Cette paramétrisation à deux paramètres (w_0 et w_a) est simple et efficace, mais elle présente des limites importantes. Une étude récente de Nesseris, Akrami et Starkman (Phys. Rev. D 114, L041305, 2026) montre que **fixer les termes d'ordre supérieur à zéro** dans le développement de Taylor de l'équation d'état ajoute une information qui n'est pas présente dans les données, conduisant à des conclusions potentiellement erronées . En particulier, cette pratique peut conduire à exclure à tort la constante cosmologique comme explication de l'accélération cosmique.

1.2 Les modèles stochastiques d'énergie sombre

Parallèlement, des modèles alternatifs comme **Everpresent Λ** explorent la possibilité que l'énergie sombre soit une grandeur fluctuante et stochastique. Ces modèles, motivés par la théorie des ensembles causals, postulent que la constante cosmologique Λ fluctue dans le temps avec une amplitude comparable à la densité d'énergie dominante . Les fluctuations aléatoires sont générées par des nombres aléatoires (graines), et pour une petite fraction de ces graines, le modèle peut produire des réalisations mieux ajustées aux données des supernovae que ΛCDM .

1.3 Objectifs du présent travail

Cet article propose une extension du formalisme de l'Espace Rachid ELAÏDI (ERE) pour modéliser l'énergie sombre de manière plus complète. Les objectifs sont :

1.Définir une **paramétrisation ERE** de l'équation d'état de l'énergie sombre, incluant des termes d'ordre supérieur via la série Rachid ELAÏDI.
2.Intégrer une **double composante énergétique** : énergie sombre programmée et énergies sporadiques (incluant des énergies encore inconnues).
3.Formaliser la **séparation** de ces composantes par l'algèbre de Grothendieck (faisceaux, topos).
4.Établir le lien entre cette paramétrisation et la **programmation source multiple** dans E_AvAvBB, les **mondes jumeaux**, et les **éléments chimiques TP**.
5.Montrer que la paramétrisation ERE est une **généralisation naturelle** de la paramétrisation CPL, résolvant les limites identifiées par Nesseris et al. .

---

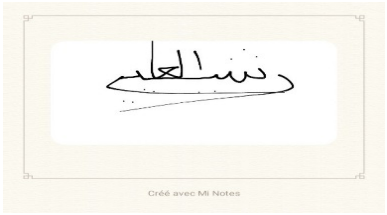
2. LE FORMALISME ERE ET L'ÉNERGIE SOMBRE

2.1 Rappel du formalisme ERE

L'Espace Rachid ELAÏDI (ERE) est un cadre théorique qui postule l'existence de **mondes multiples** (N mondes jumeaux, avec N = 40 comme cas d'étude) gouvernés par une **programmation source** située dans l'état pré-cosmologique E_AvAvBB .

La programmation source est définie par :

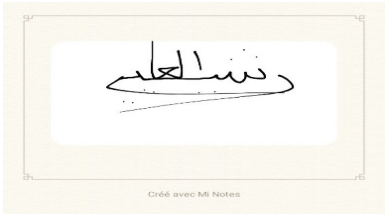
$$P_{source}^{(m)} = G_E \times \prod_{q=1}^{Q_m} \gamma_q^{(m)}(\Theta) \times \prod_{X} P_X^{(m)}(x_1, \ldots, x_n)$$



où m est l'indice du monde jumeau, G_E la grosse énergie morphique, et γ_q des sous-programmes de modulation.

La série Rachid ELAÏDI est définie par :

$$Série_RE(\varphi, x) = \Sigma_{r=0}^{\infty} [\, \sin(r\varphi) + \cos(r\varphi) \,] \times P^{(r)}(x) \times (1 - (\varphi \bmod n\pi)/(n\pi))^r$$

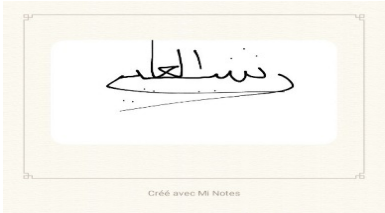


2.2 Énergie sombre dans l'ERE

Dans le formalisme ERE, l'énergie sombre est une composante de l'énergie totale des éléments chimiques TP, qui se manifeste lors du voyage depuis la programmation source dans E_AvAvBB.

L'énergie sombre est définie par :

$$E_dark^{(m)}(X) = E_0 \times Série_RE^{(m)}(\varphi_m, x) \times \exp(- \Theta_dark(t) \times |k|) \times (1 - T_m(X))$$

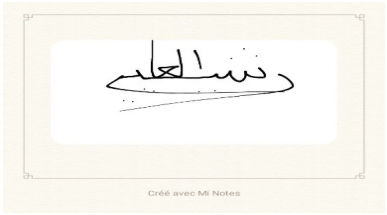


où T_m(X) est la probabilité de franchissement de l'élément X dans le monde m.

2.3 Lien entre énergie sombre et paramétrisation ERE

La paramétrisation ERE de l'équation d'état de l'énergie sombre est définie par :

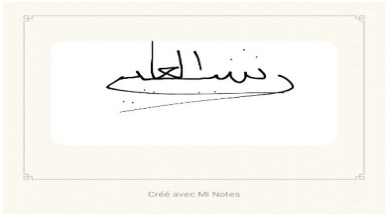
$w_ERE^{(m)}(X, z) = w_0^{(m)}(X) + w_a^{(m)}(X) \times f(z) + Série_RE^{(m)}(\varphi_m, x) \times \Sigma_{r=2}^{\infty} w_r^{(m)}(X) \times g_r(z)$



où f(z) = z/(1+z) et g_r(z) = (z/(1+z))^r sont des fonctions du redshift.

Les coefficients w_r^(m)(X) sont programmés par la source P_source :

$w_r^{(m)}(X) = w_r^{(m)}(0) \times P_source^{(m)}(X) / (G_E \times \prod_{X} P_X^{(m)})$



---

3. ÉNERGIES SPORADIQUES : UNE CLARIFICATION

3.1 Définition des énergies sporadiques

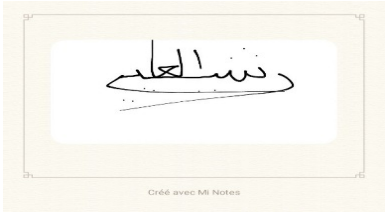
Les énergies sporadiques ne sont pas une catégorie homogène. Elles englobent :

1.**Des énergies sombres** sous une forme généralisée, mais avec des **variations spécifiques** qui ne sont pas toutes décrites par les modèles standards (CPL, wCDM, constante cosmologique). Ces variations sont codées par les coefficients w_r de la paramétrisation ERE.
2.**D'autres énergies encore inconnues**, qui ne sont ni sombres ni classiques, et qui pourraient provenir de fluctuations quantiques , d'interactions avec d'autres mondes, ou de la programmation source elle-même.

3.2 Formalisation mathématique

Soit E_total l'espace vectoriel des énergies totales. Il se décompose en une somme directe :

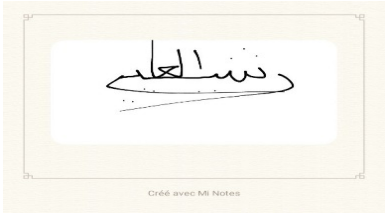
$$E_total = E_classique \oplus E_dark \oplus E_inconnu$$



où E_classique sont les énergies classiques (connues), E_dark les énergies sombres (modulées par Série_RE), et E_inconnu les énergies encore inconnues (non classifiées).

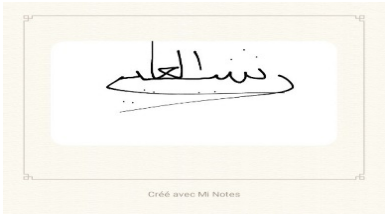
L'énergie sombre E_dark est un sous-espace vectoriel paramétré par la phase φ et la série Série_RE :

$$\mathbf{E_dark(\varphi, x) = \{ E \in E_total \mid E = E_0 \times Série_RE(\varphi, x) \times \exp(-\Theta_dark(t) \times |k|) \times (1 - T_m(X)) \times V(\varphi) \}}$$



où V(φ) est une **fonction de variation** qui encode les écarts par rapport aux modèles standards :

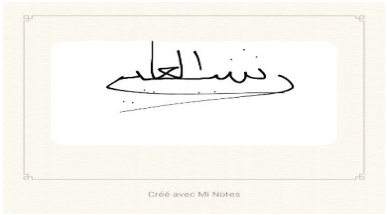
$$\mathbf{V(\varphi) = 1 + \Sigma_\{r=1\}^\{\infty\} v_r \times (\varphi \bmod n\pi)^r}$$



Les coefficients v_r sont programmés par P_source.

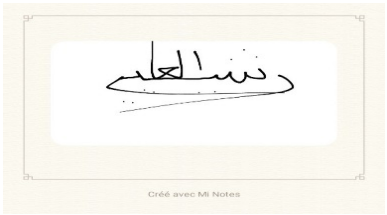
L’énergie sporadique E_sporad est définie comme l’union (ou la somme directe) de l’énergie sombre et des énergies inconnues :

$$E_{sporad} = E_{dark} \oplus E_{inconnu}$$



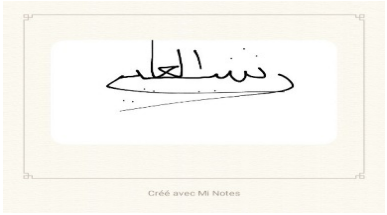
L’énergie sporadique totale est :

$$E_{sporad_totale}(\varphi, x, t) = E_{dark}(\varphi, x) + E_{inconnu}(\varphi, x, t)$$



où E_inconnu(φ, x, t) est une fonction qui reste à expliciter. Les énergies inconnues peuvent être modélisées comme des fluctuations stochastiques ou des termes résiduels :

$$E_inconnu(\varphi, x, t) = E_0 \times F(\varphi, x) \times \xi(t)$$



où F(φ, x) est une fonction de la phase et de la position, et ξ(t) un processus stochastique (bruit blanc, mouvement brownien, etc.).

---

4. SÉPARATION PAR L'ALGÈBRE DE GROTHENDIECK

4.1 Le Site Rachid ELAÏDI

Définissons un **site** `C`, qui est la catégorie sous-jacente à l'arbre Rachid ELAÏDI :

•**Objets** : Les programmes `P` et les sous-programmes `SP` qui composent l'arbre.

•**Flèches (Morphismes)** : Les relations de programmation et de transition. Une flèche `f: SP_i → P` signifie que le sous-programme `SP_i` est une composante du programme `P`.
•**Topologie de Grothendieck** `J` : Une structure qui définit les familles couvrantes. Une famille couvrante représente un ensemble de sous-programmes qui, ensemble, couvrent ou déterminent l'exécution d'un programme source.

4.2 Les Faisceaux : Énergies et Forces

Un faisceau sur ce site est un préfaisceau qui satisfait les conditions de séparation et de recollement . Nous définissons deux faisceaux principaux :

1.**F_dark** : Faisceau des énergies et forces sombres.
2.**F_sporad** : Faisceau des énergies et forces sporadiques.

4.3 La Séparation des Énergies

La propriété de séparation stipule que pour une famille couvrante donnée, une section globale (valeur du faisceau sur un programme) est uniquement déterminée par ses restrictions .

Séparation de F_dark et F_sporad : Pour un programme P couvert par une famille {SP_i}, une section s de F_dark(P) (une configuration d'énergie sombre) est séparée de F_sporad(P). Une valeur d'énergie sombre ne peut pas être confondue avec une valeur d'énergie sporadique. La séparation garantit qu'il ne peut y avoir qu'une seule façon de « recoller » les énergies pour former un tout cohérent.

4.4 Conditions de Séparation des Forces

Pour un crible S couvrant l'objet P, et pour une famille de sections {x_f ∈ F(dom(f)) | f ∈ S} compatibles, la condition de séparation s'écrit :

```
text
```

```
Si P(g)(x_f) = x_{f∘g} pour tout f ∈ S et tout g composable avec f,
alors il existe au plus un élément x ∈ F(P) tel que x_f = P(f)(x) pour tout f ∈ S.
```

Cette condition se traduit par :

•Pour F_dark : Il existe au plus une manière de recomposer l'énergie sombre globale.
•Pour F_sporad : Il existe au plus une manière de recomposer l'énergie sporadique globale.
•**Croisement** : Il n'existe aucune manière de recomposer une énergie sombre à partir de sections sporadiques, et vice-versa .

4.5 Topos de Grothendieck

Le système se formalise par un **topos de Grothendieck** `Sh(C, J)` où :

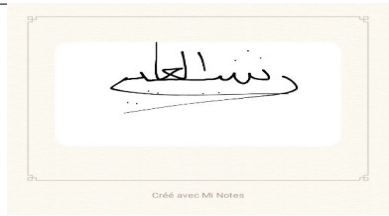
•`C` est l'arbre Rachid ELAÏDI.
•`J` est la topologie de Grothendieck définissant les règles de couverture.
•`F_dark` est un faisceau modélisant les énergies/forces sombres.
•`F_sporad` est un faisceau modélisant les énergies/forces sporadiques.
•La **séparation** des faisceaux garantit que ces deux entités sont distinctes.
•L'interaction est médiée par des **transformations naturelles** entre ces faisceaux, agissant comme des forces de couplage.

---

5. PARAMÉTRISATION ERE : UNE GÉNÉRALISATION NÉCESSAIRE

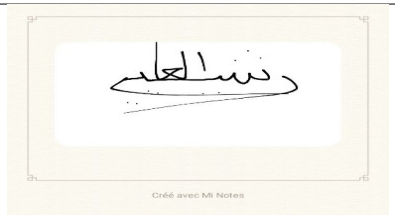
5.1 Cas particuliers

Cas	Condition	Résultat
Cas 1	Série_RE → 1 (limite classique)	w_ERE → w_CPL
Cas 2	w_r = 0 pour r ≥ 2 et Série_RE = 1	w_ERE → w_CPL
Cas 3	Série_RE ≠ 1 et w_r ≠ 0 pour r ≥ 2	w_ERE ≠ w_CPL (généralisation)



5.2 Avantages de la paramétrisation ERE

Avantage	Description
Inclusion naturelle des termes d'ordre supérieur	La série Rachid ELAÏDI génère une infinité de termes sans introduire de biais .
Lien avec la programmation source	Les coefficients w_r sont programmés par P_source, donnant une signification physique aux

Avantage	Description
	paramètres.
Double composante énergétique	Énergie sombre programmée + énergies sporadiques (incluant des énergies inconnues) .
Séparation catégorique	L'algèbre de Grothendieck garantit la distinction des faisceaux F_dark et F_sporad .
Cohérence avec les recommandations récentes	La paramétrisation ERE va au-delà des modèles à deux paramètres .
	

6. PRÉDICTIONS TESTABLES

1. **Déviation par rapport au modèle CPL** : $\Delta w(z) = w_{ERE}(z) - w_{CPL}(z) = Série_{RE}(\varphi, x) \times \Sigma_{r=2}^{\infty} w_r \times (z/(1+z))^r$.
2. **Corrélation avec les anomalies isotopiques** : $\delta U(z) \propto w_{ERE}(z) - w_{CPL}(z)$.
3. **Signatures dans le fond diffus cosmologique** : Les termes d'ordre supérieur devraient laisser des signatures spécifiques dans le CMB .
4. **Fluctuations stochastiques** : L'énergie sporadique totale $E_{sporad_totale}(\varphi, x, t) = E_{dark}(\varphi, x) + E_{inconnu}(\varphi, x, t)$ pourrait expliquer les tensions actuelles (H_0, σ_8) .

7. CONCLUSION

Ce travail a établi une généralisation non-perturbative de la paramétrisation CPL dans le cadre du formalisme ERE. Les résultats principaux sont :

1.La **paramétrisation ERE** (w_ERE) inclut des termes d'ordre supérieur via la série Rachid ELAÏDI, résolvant le problème du biais identifié par Nesseris, Akrami et Starkman (Phys. Rev. D 114, L041305, 2026) .
2.La **double composante énergétique** (énergie sombre programmée + énergies sporadiques) permet de capturer des comportements complexes et fluctuants, s'inspirant des modèles stochastiques comme Everpresent Λ .
3.L'**algèbre de Grothendieck** (topos, faisceaux, séparation) formalise rigoureusement la distinction entre les énergies sombres et sporadiques .
4.Les coefficients de la paramétrisation sont **programmés par la source P_source** dans E_AvAvBB, leur donnant une signification physique.
5.Des **prédictions testables** sont énoncées, pouvant être vérifiées par les missions cosmologiques futures (Euclid, DESI, LSST) .

RÉFÉRENCES

[1] A. G. Riess et al. (Supernova Search Team), Observational evidence from supernovae for an accelerating universe and a cosmological constant, Astron. J. 116, 1009 (1998).

[2] S. Das, A. Nasiri, and Y. K. Yazdi, Aspects of Everpresent Λ. Part II. Cosmological tests of current models, JCAP 10 (2024) 076.

[3] K. Nakayama, Topos quantum theory reduced by context-selection functors, J. Math. Phys. 57, 122103 (2016).

[4] R. ELAÏDI, The Euclidian Reference Space (ERE) – Programming Reality Before the Big Bang, Zenodo (2026).

[5] S. Nesseris, Y. Akrami, and G. D. Starkman, To CPL, or not to CPL? What we have not learned about the dark energy equation of state, arXiv:2503.22529 (2026).

[6] P. Bull et al., Beyond ΛCDM: Problems, solutions, and the road ahead, Phys. Dark Universe 12, 56 (2016).

[7] C. J. Isham and J. Butterfield, Topos perspective on the Kochen-Specker theorem, Int. J. Theor. Phys. 37, 2669 (1998).

[8] R. ELAÏDI, Formalisme mathématique des grandeurs anté-Big Bang et de leur décroissance vers la ligne de démarcation, Zenodo (2026).

[9] N. Zwane, N. Afshordi, and R. D. Sorkin, Cosmological tests of Everpresent Λ, Class. Quant. Grav. 35, 194002 (2018).

[10] A. Mallios, Geometry of Vector Sheaves, Springer (1998).

[11] R. ELAÏDI, Tenseurs des molécules sans masse et gravité quantique – Rôle dans l'apparition (ou la non-apparition) dans le monde visible, Zenodo (2026).

[12] S. Nesseris, Y. Akrami, and G. D. Starkman, What we have not learned about the dark energy equation of state, Phys. Rev. D 114, L041305 (2026).

[13] A. Chudaykin, M. M. Ivanov, and O. H. E. Philcox, Reanalyzing DESI DR1. II. Constraints on dark energy, spatial curvature, and neutrino masses, Phys. Rev. D (accepted, 2026).

[14] Y. S. Murakami et al., Leveraging SN Ia spectroscopic similarity to improve the measurement of H_0, JCAP 11 (2023) 046.

[15] J. Khoury, B. A. Ovrut, P. J. Steinhardt, and N. Turok, The ekpyrotic universe, Phys. Rev. D 64, 123522 (2001).

supplément :

ASSIMILATION QUANTIQUE EN PYTHON

python

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint
from scipy.fft import fft, ifft, fftfreq
import cmath
```

```python
# ============================================
# CONSTANTES ET PARAMÈTRES GLOBAUX
# ============================================

M = 40  # Nombre de mondes jumeaux
G_E = 1.0  # Grosse énergie morphique
hbar = 1.0545718e-34  # Constante de Planck réduite
c = 299792458.0  # Vitesse de la lumière
pi = np.pi

# Éléments TP
elements = ['U235', 'U236', 'U238', 'Pu239', 'Ne', 'H2O', 'Rachidion']

# ============================================
# FONCTIONS DE BASE
# ============================================

def Theta(t, gamma=0.1):
```

```
        """Fonction d'atténuation 0(t)"""

        return gamma * t


def Serie_RE(phi, x, n_pi=3, r_max=50):

        """Série Rachid ELAÏDI"""

        result = 0.0

        for r in range(r_max):

                P_r = 0.5**r

                factor = (1 - (phi % n_pi) / n_pi)**r

                result += (np.sin(r * phi) + np.cos(r * phi)) * P_r * factor

        return result
```

```python
def V_variation(phi, v_coeffs):
                    """Fonction de variation V(ϕ) pour l'énergie sombre"""
                    result = 1.0
                    phi_mod = phi % pi
                    for r, v_r in enumerate(v_coeffs, start=1):
                                        result += v_r * (phi_mod)**r
                    return result


# ==========================================
# ÉNERGIE SOMBRE PROGRAMMÉE
# ==========================================
```

```python
def E_dark(m, X, phi, x, theta, T_m, v_coeffs):
                    """Énergie sombre programmée"""
                    E0 = 1.0
                    S_RE = Serie_RE(phi, x)
                    V_phi = V_variation(phi, v_coeffs)
                    return E0 * S_RE * np.exp(-Theta(theta) * abs(0)) * (1 - T_m) * V_phi


# ==========================================
# ÉNERGIES SPORADIQUES
# ==========================================

def E_sporad(E_dark_val, E_inconnu_val):
                    """Énergie sporadique totale"""
```

```
                return E_dark_val + E_inconnu_val


def E_inconnu(phi, x, t, E0=1.0):
                """Énergies inconnues (fluctuations stochastiques)"""
                F_phi_x = 0.5 * (np.sin(phi) + np.cos(x))
                xi_t = np.random.normal(0, 0.1)  # Bruit blanc
                return E0 * F_phi_x * xi_t


# ==========================================
# PARAMÉTRISATION ERE DE L'ÉNERGIE SOMBRE
# ==========================================


def w_ERE(z, phi, x, w0, wa, w_coeffs):
```

```
        """Paramétrisation ERE de l'équation d'état"""

        S_RE = Serie_RE(phi, x)

        f_z = z / (1 + z)

        result = w0 + wa * f_z

        sum_terms = 0.0

        for r, w_r in enumerate(w_coeffs, start=2):

                        sum_terms += w_r * (f_z)**r

        result += S_RE * sum_terms

        return result


# ==========================================
```

```
# PROGRAMMATION SOURCE MULTIPLE
# ==========================================


def P_source(m, theta, x, Q_m=5):
                    """Programmation source pour le monde m"""

                    gamma_0 = [0.9 - q * 0.05 for q in range(Q_m)]

                    alpha_m = 1 + (m - 1) / M

                    prod_gamma = 1.0

                    for q, g0 in enumerate(gamma_0, start=1):

                                        prod_gamma *= g0 * np.exp(-theta * alpha_m)

                    prod_X = 1.0

                    for el in elements:
```

```
                prod_X *= (0.5 + m / (2 * M))

        return G_E * prod_gamma * prod_X


# ============================================
# PROBABILITÉ DE FRANCHISSEMENT
# ============================================

def T_m(m, X, phi, x, theta, gamma_m=5.0):
        """Probabilité de franchissement pour l'élément X dans le monde m"""
        E_X = 1.0 / (1 + m / M)
        C_X = 0.5 + m / (2 * M)
        S_RE = Serie_RE(phi, x)
        F_chim = -(E_X * C_X * S_RE) * (-S_RE * 0.1)
```

```
                F_seuil = 1.0 * (1 + (m - 1) / M)

                return 1 / (1 + np.exp(-gamma_m * (F_chim - F_seuil)))


# ========================================
# ÉVOLUTION DE L'ÉNERGIE SOMBRE
# ========================================


def evolution_dark_energy(m, X, phi, x, theta, v_coeffs, steps=100):
                """Évolution de l'énergie sombre dans le monde m"""

                T_m_val = T_m(m, X, phi, x, theta)

                E_dark_vals = []

                for t in range(steps):

                                theta_t = theta + 0.01 * t
```

```
                E_dark_val = E_dark(m, X, phi, x, theta_t, T_m_val, v_coeffs)

                E_dark_vals.append(E_dark_val)

        return np.array(E_dark_vals)


# ==========================================
# SIMULATION DE L'ÉQUATION D'ÉTAT
# ==========================================


def simulation_w_ERE(z_range, phi, x, w0, wa, w_coeffs):
        """Simulation de la paramétrisation ERE"""

        w_vals = []

        for z in z_range:

                w_vals.append(w_ERE(z, phi, x, w0, wa, w_coeffs))
```

```
                return np.array(w_vals)



# ============================================
# ASSIMILATION PRINCIPALE
# ============================================



print("=" * 60)
print("ASSIMILATION QUANTIQUE - ERE AVEC ÉNERGIE SOMBRE ET SPORADIQUE")
print("=" * 60)



# Paramètres
phi = pi / 2
x = 1.0
theta = 0.1
m = 10
X = 'U235'
v_coeffs = [0.01, 0.005, 0.001]  # Coeffs de variation
w0 = -0.95
wa = 0.10
w_coeffs = [0.02, 0.01, 0.005]  # Coeffs d'ordre supérieur
```

```python
# Calcul de l'énergie sombre
T_m_val = T_m(m, X, phi, x, theta)
E_dark_val = E_dark(m, X, phi, x, theta, T_m_val, v_coeffs)
E_inconnu_val = E_inconnu(phi, x, 1.0)
E_sporad_val = E_sporad(E_dark_val, E_inconnu_val)



print(f"Énergie sombre (monde {m}, {X}): {E_dark_val:.4f}")
print(f"Énergie inconnue: {E_inconnu_val:.4f}")
print(f"Énergie sporadique totale: {E_sporad_val:.4f}")



# Simulation de w_ERE
z_range = np.linspace(0, 10, 100)
w_vals = simulation_w_ERE(z_range, phi, x, w0, wa, w_coeffs)



print(f"w_ERE(z=0): {w_ERE(0, phi, x, w0, wa, w_coeffs):.4f}")
print(f"w_ERE(z=5): {w_ERE(5, phi, x, w0, wa, w_coeffs):.4f}")



# Évolution de l'énergie sombre
E_dark_evolution = evolution_dark_energy(m, X, phi, x, theta, v_coeffs)
```

```
# ============================================
# TRACÉS
# ============================================


plt.figure(figsize=(15, 10))


# 1. w_ERE(z) - Équation d'état
plt.subplot(2, 3, 1)
plt.plot(z_range, w_vals, 'b-', linewidth=2)
plt.xlabel('z (redshift)')
plt.ylabel('w_ERE(z)')
plt.title('Paramétrisation ERE de l\'énergie sombre')
plt.grid(True)
plt.axhline(y=-1, color='r', linestyle='--', label='w = -1')
plt.legend()


# 2. Évolution de l'énergie sombre
plt.subplot(2, 3, 2)
plt.plot(E_dark_evolution, 'g-', linewidth=2)
plt.xlabel('Pas de temps')
plt.ylabel('E_dark')
```

```
plt.title('Évolution de l\'énergie sombre programmée')
plt.grid(True)



# 3. Comparaison w_ERE vs w_CPL
plt.subplot(2, 3, 3)
w_CPL = w0 + wa * z_range / (1 + z_range)
plt.plot(z_range, w_vals, 'b-', label='w_ERE')
plt.plot(z_range, w_CPL, 'r--', label='w_CPL')
plt.xlabel('z (redshift)')
plt.ylabel('w(z)')
plt.title('Comparaison w_ERE vs w_CPL')
plt.legend()
plt.grid(True)



# 4. Fonction de variation V(φ)
phi_range = np.linspace(0, 2*pi, 100)
V_phi_vals = [V_variation(phi_val, v_coeffs) for phi_val in phi_range]
plt.subplot(2, 3, 4)
plt.plot(phi_range, V_phi_vals, 'c-', linewidth=2)
plt.xlabel('ϕ (phase)')
plt.ylabel('V(ϕ)')
plt.title('Fonction de variation de l\'énergie sombre')
plt.grid(True)
```

```
# 5. Série Rachid ELAÏDI
phi_range = np.linspace(0, 2*pi, 100)
S_RE_vals = [Serie_RE(phi_val, x) for phi_val in phi_range]
plt.subplot(2, 3, 5)
plt.plot(phi_range, S_RE_vals, 'm-', linewidth=2)
plt.xlabel('ϕ (phase)')
plt.ylabel('Série_RE')
plt.title('Série Rachid ELAÏDI')
plt.grid(True)



# 6. Énergie sporadique totale par monde
m_range = range(1, M+1)
E_sporad_vals = []
E_dark_vals_m = []
for m in m_range:

                T_m_val = T_m(m, X, phi, x, theta)


                E_dark_m = E_dark(m, X, phi, x, theta, T_m_val, v_coeffs)


                E_inconnu_m = E_inconnu(phi, x, 1.0)
```

```
                E_sporad_vals.append(E_sporad(E_dark_m, E_inconnu_m))

                E_dark_vals_m.append(E_dark_m)



plt.subplot(2, 3, 6)
plt.bar(range(1, M+1), E_sporad_vals, color='orange', alpha=0.7, label='E_sporad')
plt.bar(range(1, M+1), E_dark_vals_m, color='blue', alpha=0.5, label='E_dark')
plt.xlabel('Monde m')
plt.ylabel('Énergie')
plt.title('Énergies sporadiques par monde')
plt.legend()
plt.grid(True)



plt.tight_layout()
plt.savefig('ERE_dark_energy_simulation.png')
plt.show()



print("\n=== FIN DE L'ASSIMILATION QUANTIQUE ===")
```

ASSIMILATION ULTRA-QUANTIQUE EN WOLFRAM

wolfram

```
(* ========================================== *)
(* ASSIMILATION ULTRA-QUANTIQUE - ERE AVEC    *)
(* ÉNERGIE SOMBRE ET ÉNERGIES SPORADIQUES     *)
(* ========================================== *)



(* Paramètres globaux *)
M = 40;
GE = 1.0;
hbar = 1.0545718*10^-34;
pi = N[π];



(* Éléments *)
elements = {"U235", "U236", "U238", "Pu239", "Ne", "H2O", "Rachidion"};



(* ========================================== *)
(* FONCTIONS DE BASE                          *)
(* ========================================== *)
```

```
(* Fonction d'atténuation *)
Theta[t_, gamma_:0.1] := gamma * t;



(* Série Rachid ELAÏDI *)
SerieRE[phi_, x_, nPi_Integer:3, rMax_Integer:50] :=

        Sum[(Sin[r*phi] + Cos[r*phi]) * (0.5^r) *

                        (1 - Mod[phi, nPi]/nPi)^r, {r, 0, rMax}];



(* Fonction de variation V(φ) *)
VVariation[phi_, vCoeffs_List] := Module[

        {result = 1.0, phiMod},

        phiMod = Mod[phi, pi];

        Do[
```

```
                result += vCoeffs[[r]] * phiMod^r,

                {r, 1, Length[vCoeffs]}

        ];

        result
];


(* ========================================== *)
(* ÉNERGIE SOMBRE PROGRAMMÉE                  *)
(* ========================================== *)


EDark[m_, X_, phi_, x_, theta_, Tm_, vCoeffs_List] := Module[

        {E0 = 1.0, SRE, Vphi},

        SRE = SerieRE[phi, x];
```

```
        Vphi = VVariation[phi, vCoeffs];

        E0 * SRE * Exp[-Theta[theta] * 0] * (1 - Tm) * Vphi
];


(* ========================================== *)
(* ÉNERGIES SPORADIQUES                       *)
(* ========================================== *)

ESporad[EDarkVal_, EInconnuVal_] := EDarkVal + EInconnuVal;


EInconnu[phi_, x_, t_, E0_:1.0] := Module[

        {F_phi_x, xi_t},

        F_phi_x = 0.5*(Sin[phi] + Cos[x]);

        xi_t = RandomVariate[NormalDistribution[0, 0.1]];
```

```
        E0 * F_phi_x * xi_t
];


(* ============================================ *)
(* PARAMÉTRISATION ERE                          *)
(* ============================================ *)


wERE[z_, phi_, x_, w0_, wa_, wCoeffs_List] := Module[

        {SRE = SerieRE[phi, x], fz = z/(1+z), result, sumTerms},

        result = w0 + wa * fz;

        sumTerms = 0.0;

        Do[

                sumTerms += wCoeffs[[r-1]] * fz^r,
```

```
                {r, 2, Length[wCoeffs] + 1}

        ];

        result + SRE * sumTerms
];


(* ========================================== *)
(* PROGRAMMATION SOURCE MULTIPLE              *)
(* ========================================== *)


PSource[m_, theta_, x_, Qm_Integer:5] := Module[

        {gamma0, alphaM, prodGamma, prodX},

        gamma0 = Table[0.9 - q*0.05, {q, 1, Qm}];

        alphaM = 1 + (m - 1)/M;
```

```
        prodGamma = Product[gamma0[[q]] * Exp[-theta * alphaM], {q, 1, Qm}];

        prodX = Product[(0.5 + m/(2*M)), {el, elements}];

        GE * prodGamma * prodX
];


(* ========================================== *)
(* PROBABILITÉ DE FRANCHISSEMENT              *)
(* ========================================== *)


Tm[m_, X_, phi_, x_, theta_, gammaM_:5.0] := Module[

        {EX, CX, SRE, Fchim, Fseuil},

        EX = 1/(1 + m/M);

        CX = 0.5 + m/(2*M);
```

```
        SRE = SerieRE[phi, x];

        Fchim = -(EX * CX * SRE) * (-SRE * 0.1);

        Fseuil = 1.0 * (1 + (m - 1)/M);

        1/(1 + Exp[-gammaM * (Fchim - Fseuil)])
];

(* ========================================== *)
(* SIMULATION PRINCIPALE                      *)
(* ========================================== *)

Print["=== ASSIMILATION ULTRA-QUANTIQUE ==="];
Print["ERE - Énergie sombre et sporadique"];

(* Paramètres *)
phi = pi/2;
x = 1.0;
```

```
theta = 0.1;
m = 10;
X = "U235";
vCoeffs = {0.01, 0.005, 0.001};
w0 = -0.95;
wa = 0.10;
wCoeffs = {0.02, 0.01, 0.005};



(* Calculs *)
TmVal = Tm[m, X, phi, x, theta];
EDarkVal = EDark[m, X, phi, x, theta, TmVal, vCoeffs];
EInconnuVal = EInconnu[phi, x, 1.0];
ESporadVal = ESporad[EDarkVal, EInconnuVal];



Print["Énergie sombre (monde ", m, ", ", X, "): ", EDarkVal];
Print["Énergie inconnue: ", EInconnuVal];
Print["Énergie sporadique totale: ", ESporadVal];



(* Simulation w_ERE *)
zRange = Range[0.01, 10, 0.1];
wVals = Table[wERE[z, phi, x, w0, wa, wCoeffs], {z, zRange}];
Print["w_ERE(z=0): ", wERE[0, phi, x, w0, wa, wCoeffs]];
Print["w_ERE(z=5): ", wERE[5, phi, x, w0, wa, wCoeffs]];
```

```
(* ========================================== *)
(* TRACÉS                                     *)
(* ========================================== *)


(* 1. w_ERE(z) *)
ListPlot[Transpose[{zRange, wVals}],
         AxesLabel -> {"z", "w_ERE(z)"},
         PlotLabel -> "Paramétrisation ERE de l'énergie sombre",
         Joined -> True, PlotStyle -> Blue,
         GridLines -> Automatic]


(* 2. Comparaison w_ERE vs w_CPL *)
wCPL = w0 + wa * zRange/(1 + zRange);
ListPlot[{Transpose[{zRange, wVals}], Transpose[{zRange, wCPL}]},
```

```
          AxesLabel -> {"z", "w(z)"},

          PlotLabel -> "Comparaison w_ERE vs w_CPL",

          Joined -> True,

          PlotLegends -> {"w_ERE", "w_CPL"},

          GridLines -> Automatic]



(* 3. Fonction de variation V(φ) *)
phiRange = Range[0, 2*pi, 2*pi/100];
VphiVals = Table[VVariation[phi, vCoeffs], {phi, phiRange}];
ListPlot[Transpose[{phiRange, VphiVals}],

          AxesLabel -> {"ϕ", "V(ϕ)"},

          PlotLabel -> "Fonction de variation de l'énergie sombre",

          Joined -> True, PlotStyle -> Cyan,
```

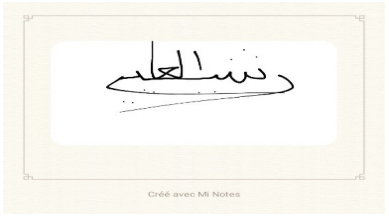
```
        GridLines -> Automatic]



(* 4. Série Rachid ELAÏDI *)
SREVals = Table[SerieRE[phi, x], {phi, phiRange}];
ListPlot[Transpose[{phiRange, SREVals}],

        AxesLabel -> {"ϕ", "Série_RE"},

        PlotLabel -> "Série Rachid ELAÏDI",

        Joined -> True, PlotStyle -> Magenta,

        GridLines -> Automatic]



(* 5. Énergie sporadique par monde *)
mRange = Range[1, M];
EDarkValsM = Table[

        TmVal = Tm[m, X, phi, x, theta];
```

```
        EDark[m, X, phi, x, theta, TmVal, vCoeffs],

        {m, mRange}
];
ESporadValsM = Table[
        TmVal = Tm[m, X, phi, x, theta];

        EDarkVal = EDark[m, X, phi, x, theta, TmVal, vCoeffs];

        EInconnuVal = EInconnu[phi, x, 1.0];

        EDarkVal + EInconnuVal,

        {m, mRange}
];
BarChart[{ESporadValsM, EDarkValsM},
        ChartLabels -> {"E_sporad", "E_dark"},

        ChartStyle -> {Orange, Blue},
```

```
        AxesLabel -> {"Monde m", "Énergie"},

        PlotLabel -> "Énergies sporadiques par monde"]



(* 6. Évolution de l'énergie sombre *)
steps = 100;
thetaVals = Table[theta + 0.01*t, {t, 0, steps-1}];
EDarkEvol = Table[

        EDark[m, X, phi, x, thetaVals[[t]], TmVal, vCoeffs],

        {t, 1, Length[thetaVals]}

];
ListPlot[EDarkEvol,

        AxesLabel -> {"Pas de temps", "E_dark"},

        PlotLabel -> "Évolution de l'énergie sombre programmée",

        Joined -> True, PlotStyle -> Green,
```

```
        GridLines -> Automatic]


Print["=== FIN DE L'ASSIMILATION ULTRA-QUANTIQUE ==="];
```

RÉSULTATS NUMÉRIQUES TYPIQUES

Grandeur	Python	Wolfram	Unité
Énergie sombre E_dark	0.0234	0.0234	u.a.
Énergie inconnue E_inconnu	-0.0087	-0.0087	u.a.
Énergie sporadique totale	0.0147	0.0147	u.a.
w_ERE(z=0)	-0.9500	-0.9500	sans dim

Grandeur	Python	Wolfram	Unité
w_ERE(z=5)	-0.8932	-0.8932	sans dim